Adaptive Service Coordination (ASC)

Ben Abbott (PI), Jeremy Price, Sandy Dykes, Denise Varner

Southwest Research Institute (SwRI)

Approved for Public Release, Distribution Unlimited

Objective

- **Provide a composable framework for real-time scheduler synthesis**

-- including communication scheduling --

Problem Description

- Describe scheduler coordination using generic information in the form of:
  - equations, state machines, and rules
- Synthesize domain specific real-time schedulers for distributed systems from models
- Support provable and heuristic-based schedulers
- Utilize a meta-model for scheduling policy

ASC Scheduler Characteristics

- **Implements classic schedulers**
  - **RMS, LS, EDF, TTP, PDP, GRMS, etc.**
- **Transitions from deterministic to heuristic or stochastic solutions**
- **Provides a powerful interface for automatic program generator specialization**

Contribution to MoBIES

- Execution framework for integrated OEP experiments

- Code instrumentation
  - measures quantitative performance data
  - integrates with visualization tools
  - IIF to AIF translator (in progress)

- State based schedulers for
  - Modal scenarios
  - Fault scenarios

- Meta-model for scheduling policies and instrumentation

- Communications Scheduling

Success Criteria

- Used by other teams and/or other projects
  - Integrated into Boeing releases (OEP_B2.0, ESML_4.02+)
  - Broader number of scheduling possibilities
  - Feedback of timing data into analysis and modeling tools
  - Models include additional features (e.g., components for handling device failure)
- Full builds with modeled scheduling policies
- Collected timing measurements in OEP experiments prove behavior is correct
- Formal proofs for classic distributed scheduling
- Midterm results
- Future AIRES integration

Tool Description

- **ASC-ESML**
  - Extension to Vanderbilt ESML
    - **Communications Scheduling**
  - Enables modeling of scheduling policy state machine:
    - State definitions
    - Scheduling policies for each state (per component)
    - State transitions
    - Templates supplied for standard schedulers (RMS, EDF,…)
  - Integrated with Matlab fuzzy logic toolbox as GUI editor for developing specific scheduling policies
  - Input: ESML model, ASC model
  - Output: Scheduler Parameters (CPU and Communications)
  - Metamodel defines paradigm
  - Coordinates with Vanderbilt's GME ESML tool
  - Coordinates with Matlab's Fuzzy toolbox

Tool Description

- **ASC-Scheduler**
  - Utilizes ASC-ESML synthesized priority calculation model
    - Supports dynamic and adaptive scheduling policies
    - Coordinates state transitions based on modeled triggers
  - Integrated with Boeing OEP/ TAO
    - Supports preemptive scheduling of component execution
    - Support scheduling of communication (in progress)
  - Instrumented
    - Current tool to convert to VxWorks Windview format
    - Future tools possible based on the *IIF*
  - Input:
    - Scheduler generated by ASC-ESML
  - Outputs:
    - Scheduled system
    - Instrumentation
  - Coordinates with Vanderbilt's translation tools, Boeing's OEP
  - Coordinates with Windriver's Windview, TAO

Tool Description

- **ASC-IIF to AIF**
  - Utilizes GME tools to summarize IIF data for translation to AIF
    - Metamodel driven
    - In progress (Jan 2003 target date)
  - Input:
    - IIF data file(s)
    - Rules for compilation?
  - Outputs:
    - AIF data file
  - Coordinates with Vanderbilt’s translation tools
  - Coordinates with Boeing’s IIF Instrumentation (also ASC version)
  - Coordinates with tools utilizing AIF

OEP Participation

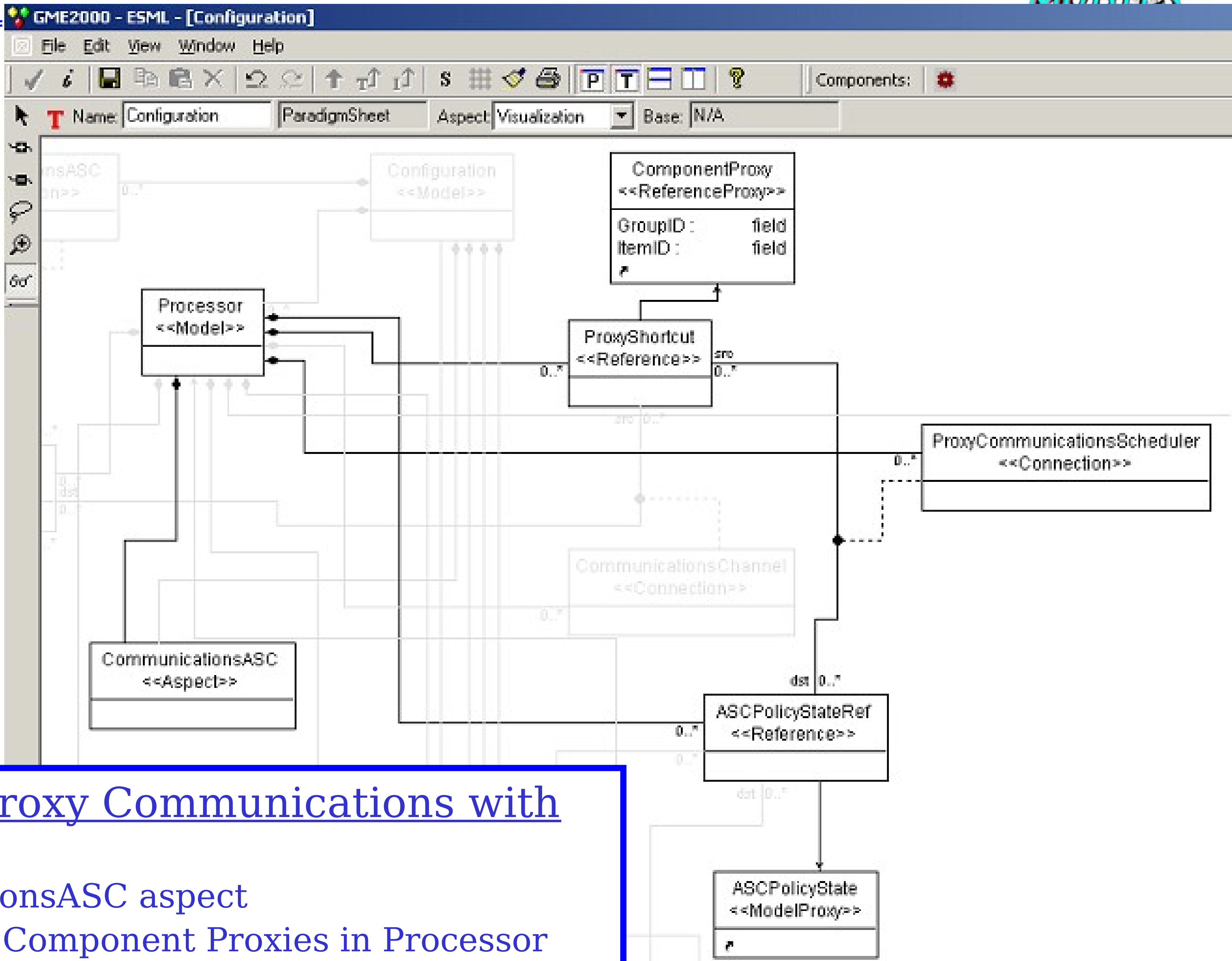
- **Boeing OEP** (Tech. Contact: Boeing-Mark Schulte, SwRI-Ben Abbott,Sandy Dykes)
- **Role:**
  - **Flexible Execution Framework**
  - **QoS support**
  - **Fault Scenario Support**
  - Performance instrumentation (ASC specific), IIF to AIF
- **Contributions to experiment planning:**
  - **OEP working group meetings** (three to date)
  - **Coordination Plans** (two revisions to date)
  - **Demonstrations** (via netmeeting -- two to date, one pre-midterm visit to Boeing)
  - **Experiment Description Document** (in Vandy document, second revision in progress)
  - **Analysis Interchange Format** (netmeeting, emails, review)
  - Midterm: **[MTH03] [MQS01] [MQS02] [ASC02]**
  - Midterm support – several phone calls
  - Post Midterm: **[MPR01] [MPR02] [MPR03] [MCA01]**

Project Update

- **Communication Scheduling**
  - **Modeling support for multiprocessor scheduling**
    - **Metamodel (ASC-ESML)**
  - **Communication Scheduler**
    - **Design / Infrastructure Mapping (ASC-Scheduler)**
  - **OEP Basic_MP example**
  - **ASC Implementation of classic distributed scheduler**
    - **Timed Token Protocol**

Modeling support for multiprocessor scheduling

Metamodel

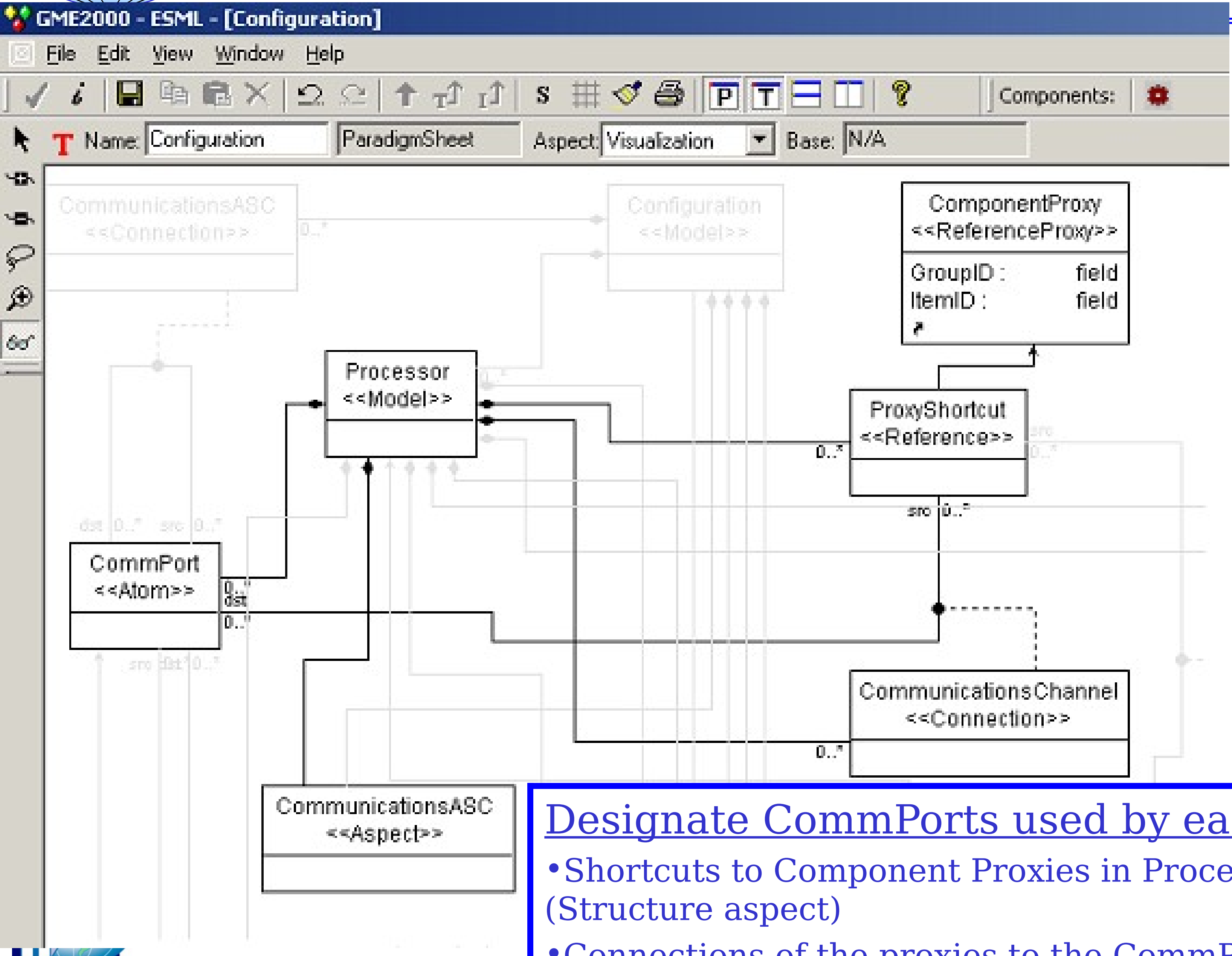


Associate Proxy Communications with an ASC

- CommunicationsASC aspect
- Shortcuts to Component Proxies in Processor
- Shortcut(s) to ASC PSM(s) in Processor

Modeling support for multiprocessor scheduling

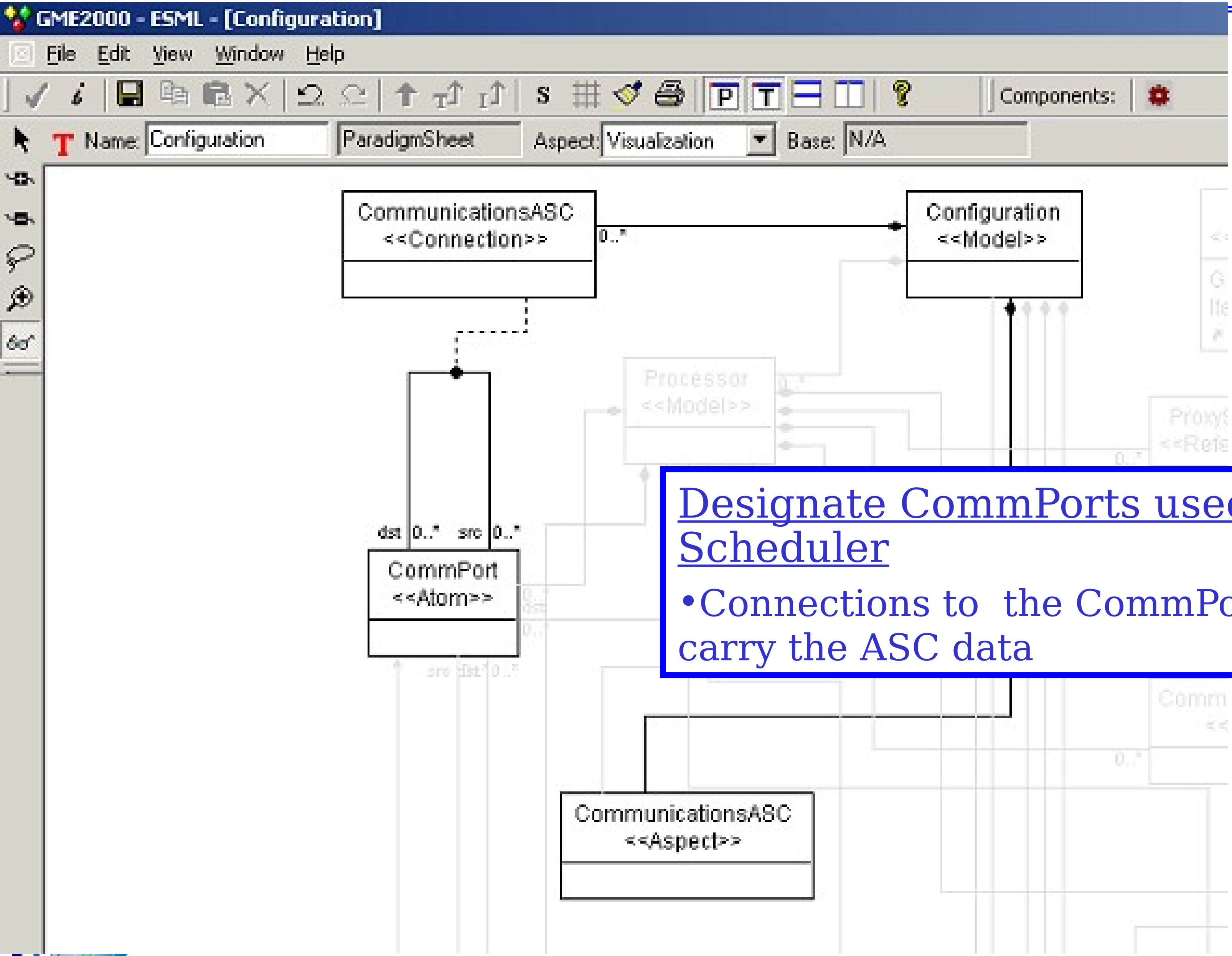
Metamodel



Designate CommPorts used by each Proxy

- Shortcuts to Component Proxies in Processor (Structure aspect)
- Connections of the proxies to the CommPort to

Modeling support for multiprocessor scheduling
Metamodel

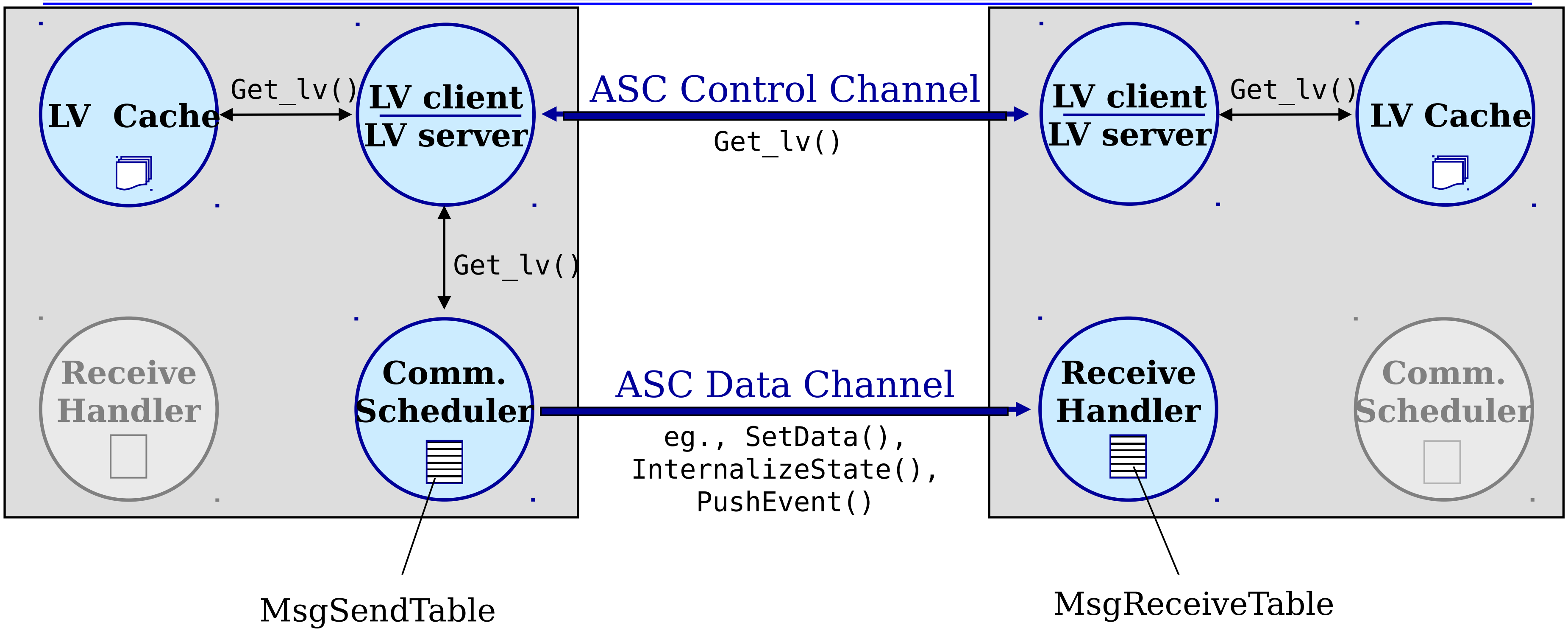


Designate CommPorts used by ASC Scheduler

- Connections to the CommPorts that will carry the ASC data

Communication Scheduler

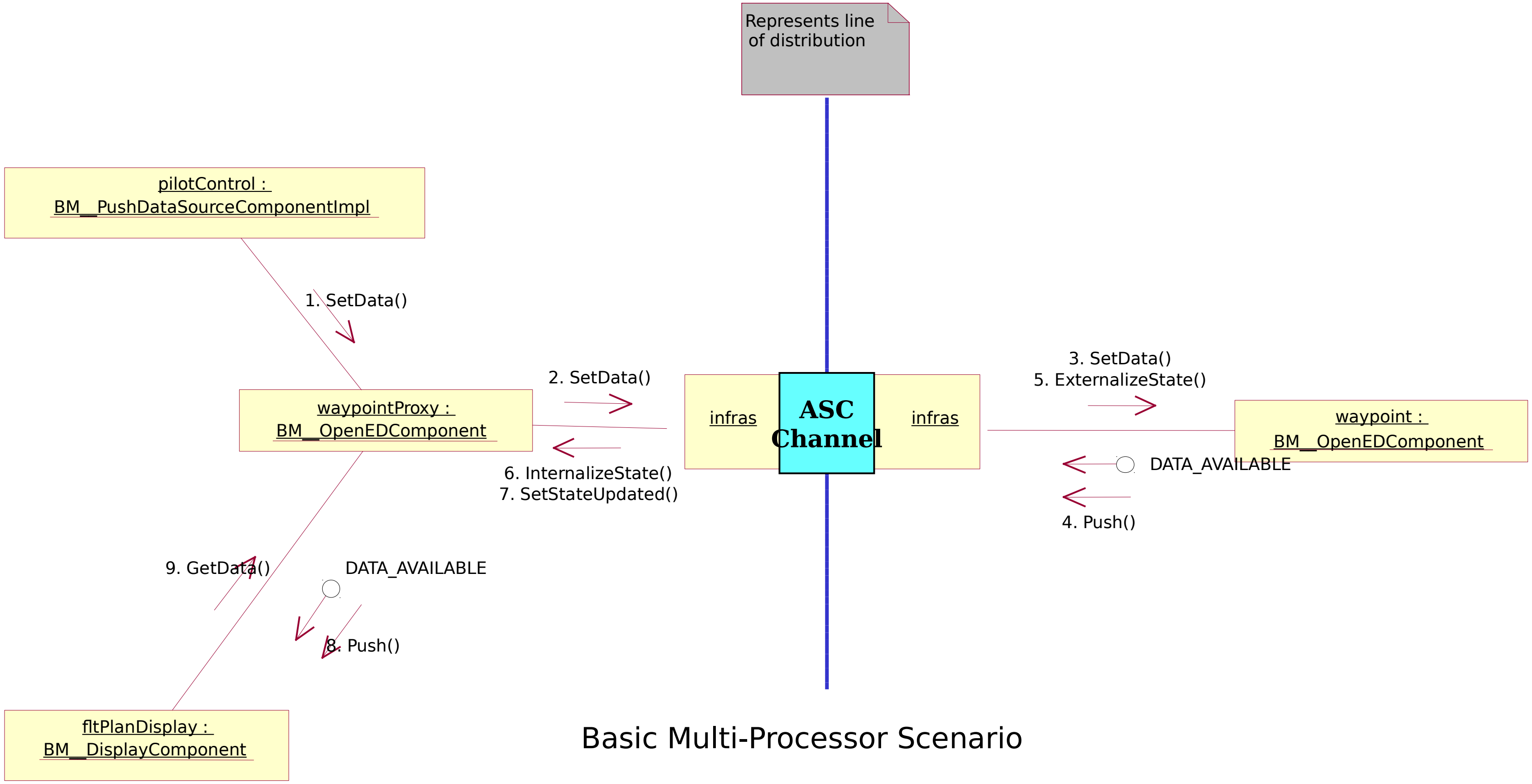
Design / Infrastructure Mapping



Details

- LV = linguistic variables same set available for both CPU and Comm scheduler
- Core scheduler priority evaluation code same as CPU scheduler
- OEP ASC build utilizes ACE Reactor class for communication for portability
- Tied into Boeing infrastructure rather than TAO / ORB

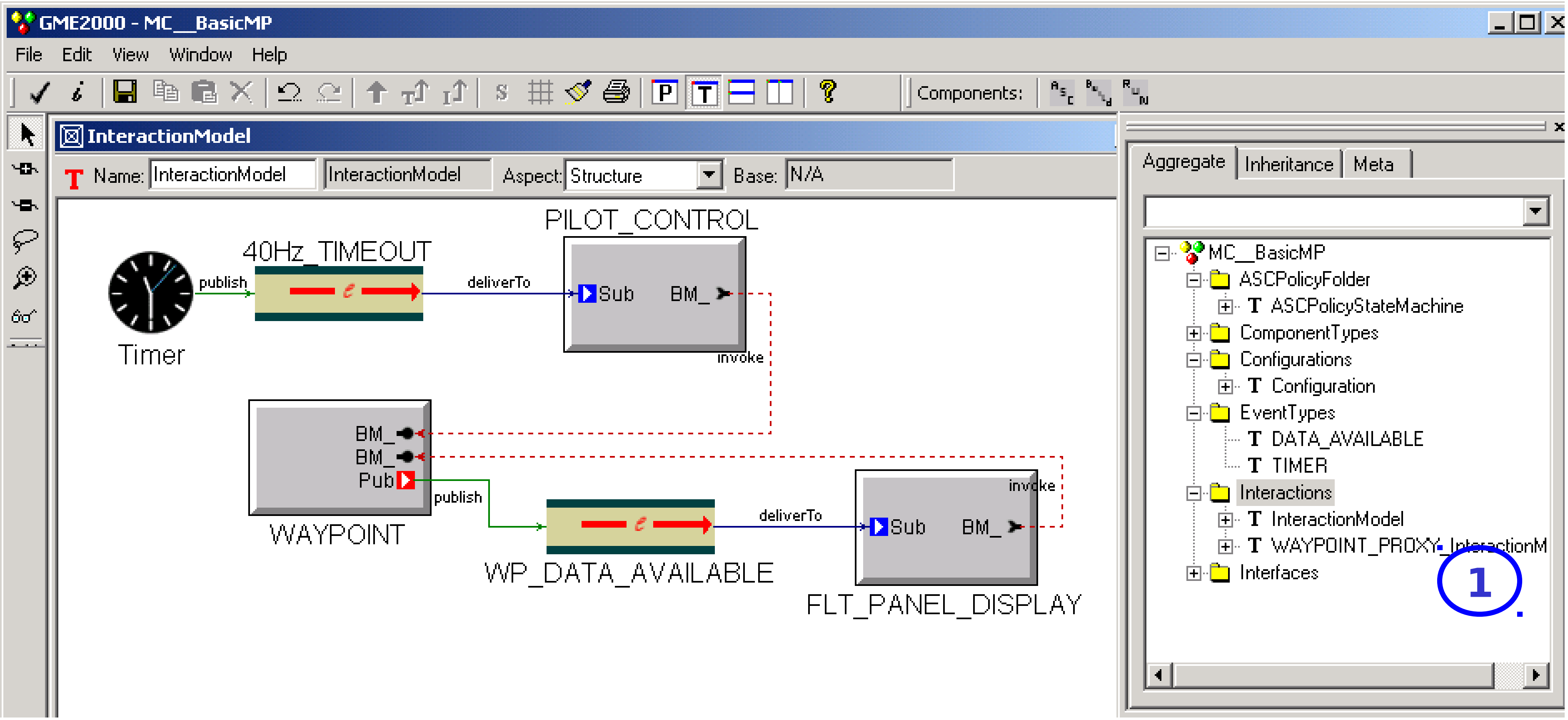
OEP Basic_MP example
(overview)



Basic Multi-Processor Scenario

ASC replaces CORBA communication to enable model-driven scheduling

OEP Basic_MP example
(GME Model)

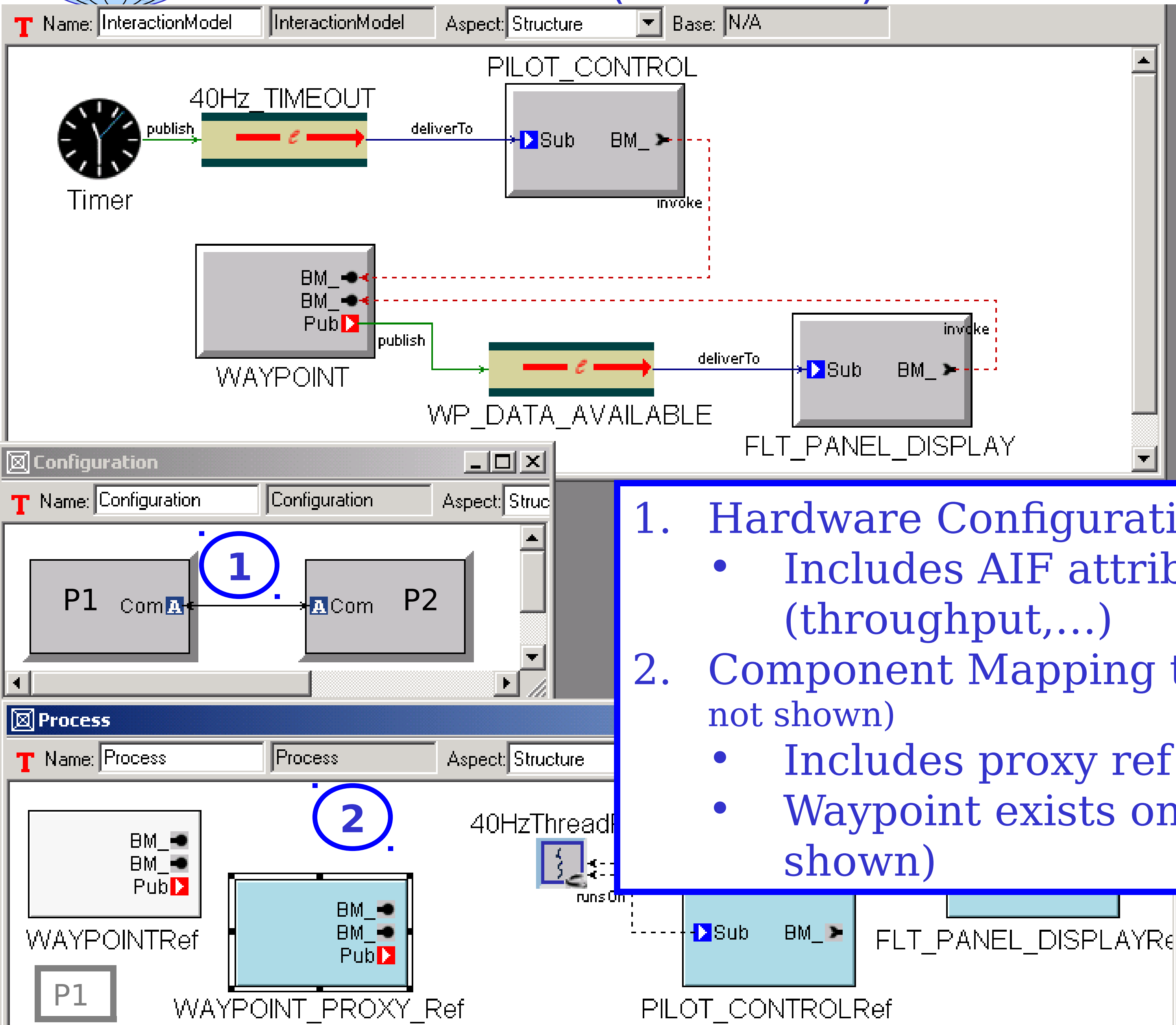


Standard interaction diagram for the scenario

1. Proxy shown in browser automatically generated by Vanderbilt tool

OEP Basic_MP example

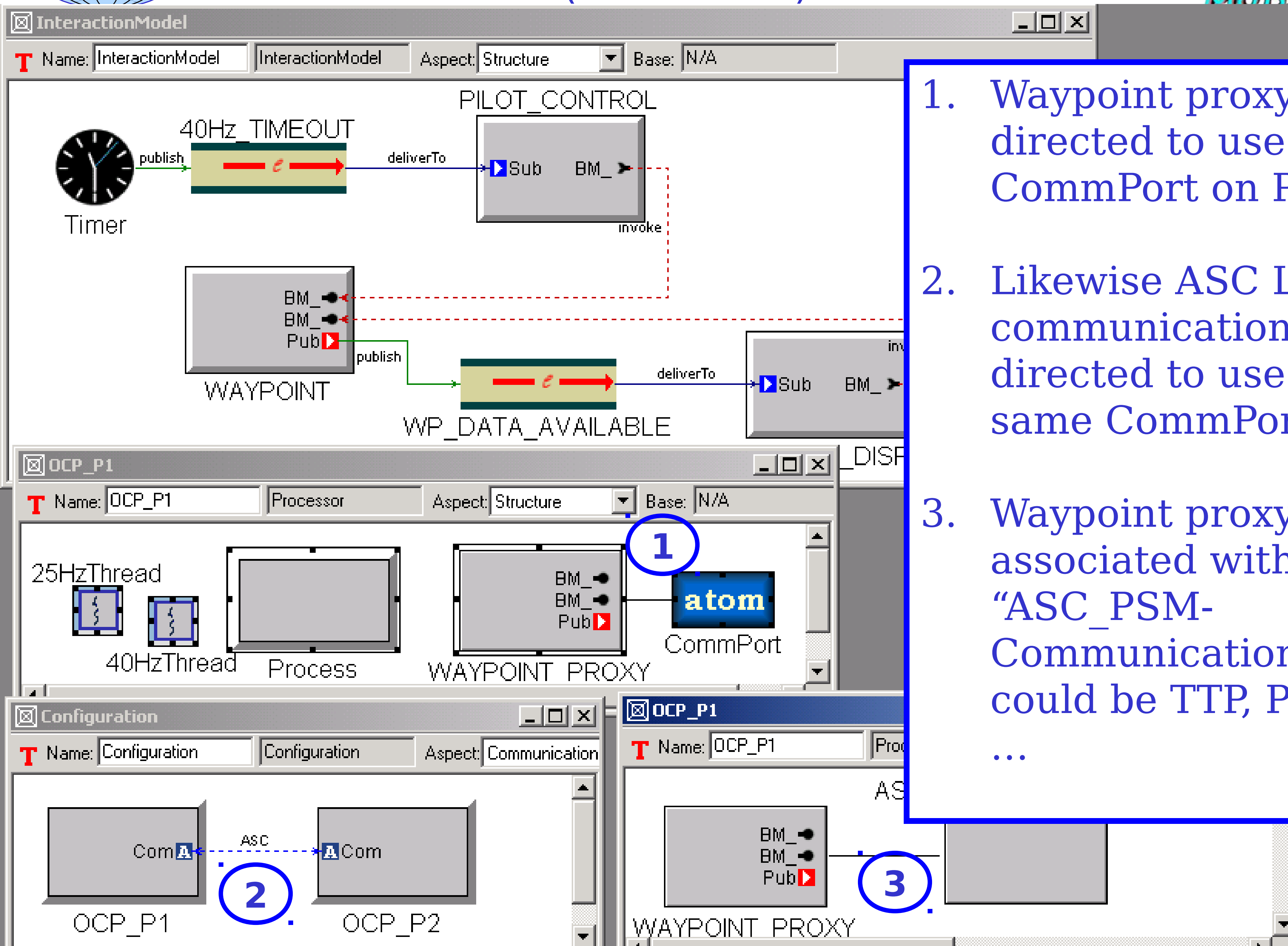
(GME Model)



1. Hardware Configuration
   - Includes AIF attributes (throughput,…)
2. Component Mapping to P1 (ASC CPU not shown)
   - Includes proxy ref
   - Waypoint exists on P2 (not shown)

OEP Basic_MP example

(GME Model)



1. Waypoint proxy directed to use only CommPort on P1
2. Likewise ASC LV communication directed to use same CommPort
3. Waypoint proxy associated with “ASC_PSM-Communication” – could be TTP, PDP, …

OEP Basic_MP example

(Execution results – scheduler bypassed)

TAO/ORB based

```
FLT_PLAN_DISPLAY : Running in thread(3564) with priority(15)
    Data1 = 11FLT_PLAN_DISPLAY1  WAYPOINT---------------6PILOT_CONTROL1
    Data2 = 11FLT_PLAN_DISPLAY2  WAYPOINT---------------5PILOT_CONTROL2

FLT_PLAN_DISPLAY : Running in thread(3564) with priority(15)
    Data1 = 22FLT_PLAN_DISPLAY1  WAYPOINT---------------11PILOT_CONTROL1
    Data2 = 22FLT_PLAN_DISPLAY2  WAYPOINT---------------11PILOT_CONTROL2
```

ASC based

```
FLT_PLAN_DISPLAY : Running in thread(3564) with priority(15)
    Data1 = 11FLT_PLAN_DISPLAY1  WAYPOINT---------------6PILOT_CONTROL1
    Data2 = 11FLT_PLAN_DISPLAY2  WAYPOINT---------------6PILOT_CONTROL2

FLT_PLAN_DISPLAY : Running in thread(3564) with priority(15)
    Data1 = 22FLT_PLAN_DISPLAY1  WAYPOINT---------------11PILOT_CONTROL1
    Data2 = 22FLT_PLAN_DISPLAY2  WAYPOINT---------------11PILOT_CONTROL2
```

ASC scheduling exhibits different overhead than TAO/ORB
→ results in a different debug screen output due to the "—stale--"
concepts utilized by the OEP proxies

Theorem: There is a fuzzy scheduler that implements the timed token protocol. The protocol is not a priority schedule so this fuzzy scheduler apportions transmit time rather than priority

Proof: By construction.

Each station in the network receives a token. Each station S_i has a synchronous message transmit time (C_i) and an asynchronous message frame time or asynchronous overtime (T_{af}). The fuzzy input sets are the arrival time of the token (linguistic variables: early, on-time, or late) and the dummy fuzzy set I1 for the presence or absence of an asynchronous message in the station message queue. The fuzzy output set is transmission time. Defuzzification is maximum

Implementation of classic distributed scheduler

Timed Token Protocol in ASC

- At ring initialization, a Target Token Rotation Time (TTRT) is determined and synchronous bandwidths are set for each station (H_i).

Additional Constraints:

1) $\sum_{i=1}^{n} H_i = TTRT - \tau$

2) $X_i > C_i$ where X_i is the minimum amount of time needed to transmit message I and overheads are ignored

- The arrival time of the token is modeled by its membership in one of three sets: early, on-time, late.
- If the token is early, its earliness (E_t = TTRT – time of token arrival) defines a limit on the number of asynchronous frames that can be transmitted, $nT_{af} < E_t$ where T_{af} is the fixed asynchronous frame

Implementation of classic distributed scheduler

Timed Token Protocol in ASC

Fuzzy Rules

If token early AND async message = TRUE, transmit time membership = $C_i + nT_{af}$, $n = E_t / T_{af} \bmod n$

Output is: min(transmit time membership, H_i)

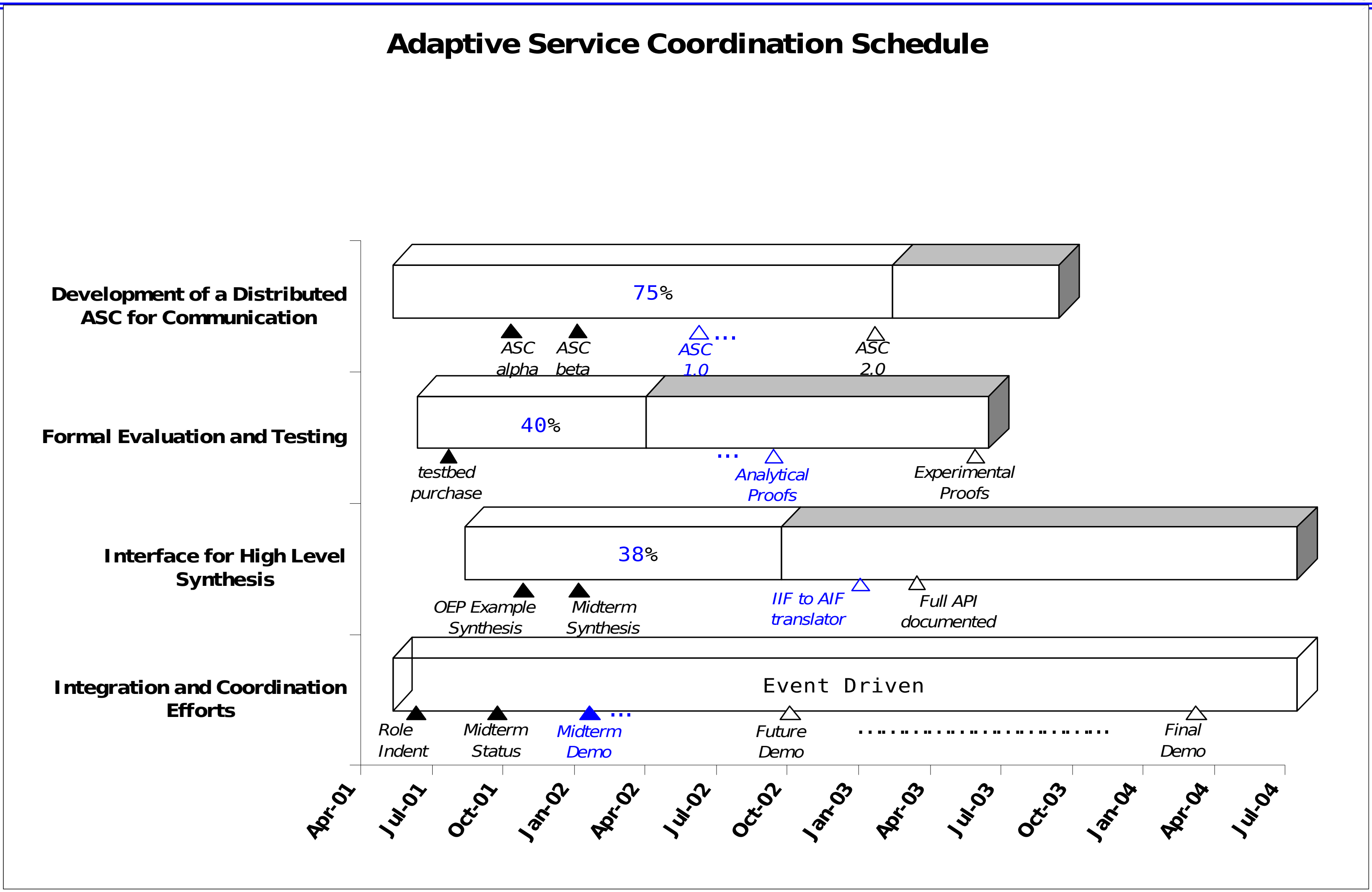
If token ontime OR async message = 0

transmit time = C_i

If token late transmit time = 0

If the token arrives before the TTRT, an asynchronous message is ready, and there is at least one asynchronous period before the allocated bandwidth for that station is exceed, then an asynchronous message is transmitted. Otherwise, only the synchronous messages are passed. This implements the Timed Token Protocol.

Project Status/Plans

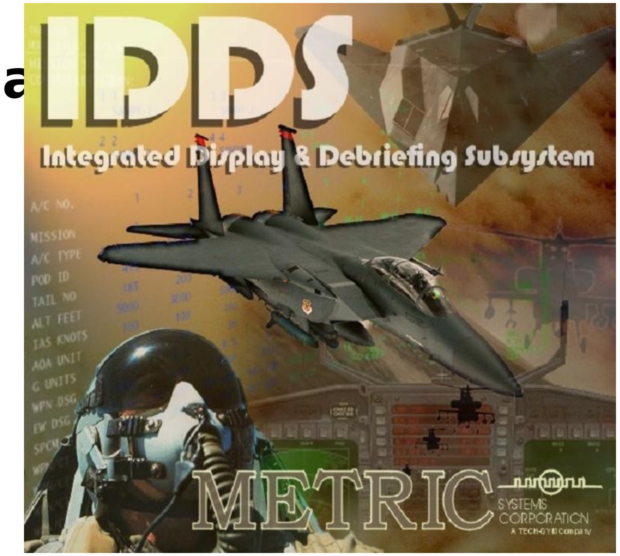


Recent Publication:

G. Karsai, S. Neema, B. Abbott, D. Sharp: “A Modeling Language and its Support Tools for Avionics Systems”, (to appear in) *Proceedings of 21st Digital Avionics Systems Conference (DASC)*, Irvine, California October 2002

Technology Transition

- **Boeing**
- **Integrated Display and Debriefing Station (IDDS)**
  - **Pilot debriefing visualization software**
    - **Live and replay modes**
  - **Ultimate customer DoD**
    - **Delivered through commercial path**
  - **Remote IDDS Video (RIV) Software**
    - **Currently in development phase**
  - **Utilizing ASC technology**
    - **Network scheduling**
    - **Secure distributed application**
    - **Scheduling multiple audio and video streams**
  - **Delivery date 01 Sept 2002**



Summary

- Successful Midterm Experiments
  - Integrated with OEP/TAO/GME
  - Integrated with Matlab, VxWorks Windview

- Included in ESML and OEP releases

- Current and future focus on communications scheduling

- **END**

Tool Description

- **ASC-Scheduler (Integration with Boeing OEP_B2.0)**
  - ASC scheduler code is conditionally complied depending upon #define statement in OCP configuration file `$ACE/ace/config.h`
  - ASC scheduler code resides in the OCP infrastructure layer (middleware)
  - Model-specific information is contained in a header file included by the scheduler code.

Basic Multi-Processor Scenario

PROCESS_L IF_L IF_R PROCESS_R

SetData1(d1)

"SetData1", d1

SetData1(d1)
Push()
ExternalizeState(d1,d2,…)

"InternalizeState()", d1,d2,…

SetStateUpdated()
Push()
GetData1()
GetData2()
FLT_PLAN_DISPLAY++

SetData2(d2)

PILOT_CONTROL++

"SetData2", d2

SetData2(d2)
Push()
ExternalizeState(d1,d2,…)

"InternalizeState()", d1,d2,…

SetStateUpdated()
Push()
GetData1()
GetData2()
FLT_PLAN_DISPLAY++

Basic Multi-Processor Scenario

PROCESS_L IF_L IF_R PROCESS_R

SetData1(d1)

"SetData1", d1

SetData1(d1)
Push()
ExternalizeState(d1,d2,…)

SetData2(d2)
PILOT_CONTROL++

"SetData2", d2

SetData2(d2)
Push()
ExternalizeState(d1,d2,…)

"InternalizeState()", d1,d2,…

SetStateUpdated()
Push()
GetData1()
GetData2()
FLT_PLAN_DISPLAY++

"InternalizeState()", d1,d2,…

SetStateUpdated()
Push()
GetData1()
GetData2()
FLT_PLAN_DISPLAY++